Basic Security Theorem

• Define action, secure formally
  – Using a bit of foreshadowing for “secure”
• Restate properties formally
  – Simple security condition
  – *-property
  – Discretionary security property
• State conditions for properties to hold
• State Basic Security Theorem

Action

• A request and decision that causes the system to move from one state to another
  – Final state may be the same as initial state
• \((r, d, v, v') \in R \times D \times V \times V\) is an action of \(\Sigma(R, D, W, z_0)\) iff there is an \((x, y, z) \in \Sigma(R, D, W, z_0)\) and a \(t \in N\) such that \((r, d, v, v') = (x_t, y_t, z_t, z_{t-1})\)
  – Request \(r\) made when system in state \(v\); decision \(d\) moves system into (possibly the same) state \(v'\)
  – Correspondence with \((x_t, y_t, z_t, z_{t-1})\) makes states, requests, part of a sequence
Simple Security Condition

- \((s, o, p) \in S \times O \times P\) satisfies the simple security condition relative to \(f\) (written \(ssc\ rel f\)) iff one of the following holds:
  1. \(p = e\) or \(p = a\)
  2. \(p = r\) or \(p = w\) and \(f(s)\ dom f(o)\)
- Holds vacuously if rights do not involve reading
- If all elements of \(b\) satisfy \(ssc\ rel f\), then state satisfies simple security condition
- If all states satisfy simple security condition, system satisfies simple security condition

Necessary and Sufficient

- \(\Sigma(R, D, W, z_0)\) satisfies the simple security condition for any secure state \(z_0\) iff for every action \((r, d, (b, m, f, h), (b', m', f', h'))\), \(W\) satisfies:
  - Every \((s, o, p) \in b - b'\) satisfies \(ssc\ rel f\)
  - Every \((s, o, p) \in b'\) that does not satisfy \(ssc\ rel f\) is not in \(b\)
- Note: “secure” means \(z_0\) satisfies \(ssc\ rel f\)
- First says every \((s, o, p)\) added satisfies \(ssc\ rel f\); second says any \((s, o, p)\) in \(b'\) that does not satisfy \(ssc\ rel f\) is deleted
*-Property

• $b(s; p_1, \ldots, p_n)$ set of all objects that $s$ has $p_1, \ldots, p_n$ access to
• State $(b, m, f, h)$ satisfies the *-property iff for each $s \in S$ the following hold:
  1. $b(s; a) \neq \emptyset \Rightarrow \forall o \in b(s; a) \left[ f_s(o) \text{ dom } f_s(s) \right]$
  2. $b(s; w) \neq \emptyset \Rightarrow \forall o \in b(s; w) \left[ f_s(o) = f_s(s) \right]$
  3. $b(s; r) \neq \emptyset \Rightarrow \forall o \in b(s; r) \left[ f_s(s) \text{ dom } f_s(o) \right]$
• Idea: for writing, object dominates subject; for reading, subject dominates object

*-Property

• If all states satisfy simple security condition, system satisfies simple security condition
• If a subset $S'$ of subjects satisfy *-property, then *-property satisfied relative to $S' \subseteq \tilde{S}$
• Note: tempting to conclude that *-property includes simple security condition, but this is false
  – See condition placed on $w$ right for each
Necessary and Sufficient

- $\Sigma(R, D, W, z_0)$ satisfies the $*$-property relative to $S' \subseteq S$ for any secure state $z_0$ iff for every action $(r, d, (b, m, f, h), (b', m', f', h'))$, $W$ satisfies the following for every $s \in S'$:
  - Every $(s, o, p) \in b - b'$ satisfies the $*$-property relative to $S'$
  - Every $(s, o, p) \in b'$ that does not satisfy the $*$-property relative to $S'$ is not in $b$
- Note: “secure” means $z_0$ satisfies $*$-property relative to $S'$
- First says every $(s, o, p)$ added satisfies the $*$-property relative to $S'$; second says any $(s, o, p)$ in $b'$ that does not satisfy the $*$-property relative to $S'$ is deleted

Discretionary Security Property

- State $(b, m, f, h)$ satisfies the discretionary security property iff, for each $(s, o, p) \in b$, then $p \in m[s, o]$
- Idea: if $s$ can read $o$, then it must have rights to do so in the access control matrix $m$
- This is the discretionary access control part of the model
  - The other two properties are the mandatory access control parts of the model
Necessary and Sufficient

• $\Sigma(R, D, W, z_0)$ satisfies the ds-property for any secure state $z_0$ iff, for every action $(r, d, (b, m, f, h), (b', m', f', h'))$, $W$ satisfies:
  - Every $(s, o, p) \in b - b'$ satisfies the ds-property
  - Every $(s, o, p) \in b'$ that does not satisfy the ds-property is not in $b$

• Note: “secure” means $z_0$ satisfies ds-property
• First says every $(s, o, p)$ added satisfies the ds-property; second says any $(s, o, p)$ in $b'$ that does not satisfy the *-property is deleted

Secure

• A system is secure iff it satisfies:
  - Simple security condition
  - *-property
  - Discretionary security property
• A state meeting these three properties is also said to be secure
Basic Security Theorem

• \( \Sigma(R, D, W, z_0) \) is a secure system if \( z_0 \) is a secure state and \( W \) satisfies the conditions for the preceding three theorems
  – The theorems are on the slides titled “Necessary and Sufficient”

Rule

• \( \rho: R \times V \rightarrow D \times V \)
• Takes a state and a request, returns a decision and a (possibly new) state
• Rule \( \rho \) ssc-preserving if for all \((r, v) \in R \times V \) and \( v \) satisfying \( ssc \ rel \ f \), \( \rho(r, v) = (d, v') \) means that \( v' \) satisfies \( ssc \ rel \ f' \).
  – Similar definitions for \( * \)-property, ds-property
  – If rule meets all 3 conditions, it is security-preserving
Unambiguous Rule Selection

- Problem: multiple rules may apply to a request in a state
  - if two rules act on a read request in state \( v \) …
- Solution: define relation \( W(\omega) \) for a set of rules \( \omega = \{ \rho_1, \ldots, \rho_m \} \) such that a state \( (r, d, v, v') \in W(\omega) \) iff either
  - \( d = i \); or
  - for exactly one integer \( j \), \( \rho_j(r, v) = (d, v') \)
- Either request is illegal, or only one rule applies

Rules Preserving SSC

- Let \( \omega \) be set of ssc-preserving rules. Let state \( z_0 \) satisfy simple security condition. Then \( \Sigma(R, D, W(\omega), z_0) \) satisfies simple security condition
  - Proof: by contradiction.
    - Choose \( (x, y, z) \in \Sigma(R, D, W(\omega), z_0) \) as state not satisfying simple security condition; then choose \( t \in \mathbb{N} \) such that \( (x_t, y_t, z_t) \) is first appearance not meeting simple security condition
    - As \( (x_t, y_t, z_t, z_{t+1}) \in W(\omega) \), there is unique rule \( \rho \in \omega \) such that \( \rho(x_t, z_{t+1}) = (y_t, z_t) \) and \( y_t \neq i \).
    - As \( \rho \) ssc-preserving, and \( z_{t+1} \) satisfies simple security condition, then \( z_t \) meets simple security condition, contradiction.
Adding States Preserving SSC

- Let \( v = (b, m, f, h) \) satisfy simple security condition. Let \( (s, o, p) \notin b \), \( b' = b \cup \{ (s, o, p) \} \), and \( v' = (b', m, f, h) \). Then \( v' \) satisfies simple security condition iff:
  1. Either \( p = e \) or \( p = a \); or
  2. Either \( p = r \) or \( p = w \), and \( f_c(s) \) dom \( f_o(o) \)

Proof

1. Immediate from definition of simple security condition and \( v' \) satisfying \( ssc \ rel \ f \)
2. \( v' \) satisfies simple security condition means \( f_c(s) \) dom \( f_o(o) \), and for converse, \( (s, o, p) \in b' \) satisfies \( ssc \ rel \ f \), so \( v' \) satisfies simple security condition

Rules, States Preserving *-Property

- Let \( \omega \) be set of *-property-preserving rules, state \( z_0 \) satisfies *-property. Then \( \Sigma(R, D, W(\omega), z_0) \) satisfies *-property
- Let \( v = (b, m, f, h) \) satisfy *-property. Let \( (s, o, p) \notin b \), \( b' = b \cup \{ (s, o, p) \} \), and \( v' = (b', m, f, h) \). Then \( v' \) satisfies *-property iff one of the following holds:
  1. \( p = e \) or \( p = a \)
  2. \( p = r \) or \( p = w \) and \( f_c(s) \) dom \( f_o(o) \)
Rules, States Preserving ds-Property

- Let $\omega$ be set of ds-property-preserving rules, state $z_0$ satisfies ds-property. Then $\Sigma(R, D, W(\omega), z_0)$ satisfies ds-property
- Let $v = (b, m, f, h)$ satisfy ds-property. Let $(s, o, p) \notin b$, $b' = b \cup \{ (s, o, p) \}$, and $v' = (b', m, f, h)$. Then $v'$ satisfies ds-property iff $p \in m[s, o]$.

Combining

- Let $\rho$ be a rule and $\rho(r, v) = (d, v')$, where $v = (b, m, f, h)$ and $v' = (b', m', f', h')$. Then:
  1. If $b' \subseteq b, f' = f$, and $v$ satisfies the simple security condition, then $v'$ satisfies the simple security condition
  2. If $b' \subseteq b, f' = f$, and $v$ satisfies the $^*$-property, then $v'$ satisfies the $^*$-property
  3. If $b' \subseteq b, m[s, o] \subseteq m'[s, o]$ for all $s \in S$ and $o \in O$, and $v$ satisfies the ds-property, then $v'$ satisfies the ds-property
Proof

1. Suppose \( v \) satisfies simple security property.
   a) \( b' \subseteq b \) and \( (s, o, r) \in b' \) implies \( (s, o, r) \in b \)
   b) \( b' \subseteq b \) and \( (s, o, w) \in b' \) implies \( (s, o, w) \in b \)
   c) So \( f_i(s) \ dom f_i(o) \)
   d) But \( f' = f \)
   e) Hence \( f'_i(s) \ dom f'_i(o) \)
   f) So \( v' \) satisfies simple security condition

2, 3 proved similarly

Example Instantiation: Multics

- 11 rules affect rights:
  - set to request, release access
  - set to give, remove access to different subject
  - set to create, reclassify objects
  - set to remove objects
  - set to change subject security level

- Set of “trusted” subjects \( S_T \subseteq S \)
  - *-property not enforced; subjects trusted not to violate

- \( \Delta(\rho) \) domain
  - determines if components of request are valid
**get-read Rule**

- Request \( r = (\text{get}, s, o, \bar{r}) \)
  - \( s \) gets (requests) the right to read \( o \)
- Rule is \( \rho_1(r, v) \):
  
  \[
  \begin{align*}
  \text{if} \ (r \neq \Delta(\rho_1)) \ &\text{then} \ \rho_1(r, v) = (\bar{r}, v); \\
  \text{else if} \ (f_1(s) \ \text{dom} \ f_1(o) \ \text{and} \ [s \in S_f \ \text{or} \ f_1(s) \ \text{dom} \ f_1(o)]) \ &\text{and} \ r \in m[s, o]) \\
  \ &\text{then} \ \rho_1(r, v) = (y, (b \cup \{ (s, o, \bar{r}) \}, m, f, h)); \\
  \ &\text{else} \ \rho_1(r, v) = (\bar{n}, v);
  \end{align*}
  \]

**Security of Rule**

- The get-read rule preserves the simple security condition, the \(*\)-property, and the \(ds\)-property
  - Proof
    - Let \( v \) satisfy all conditions. Let \( \rho_1(r, v) = (d, v') \). If \( v' = v \) result is trivial. So let \( v' = (b \cup \{ (s_2, o, \bar{r}) \}, m, f, h) \).
Proof

• Consider the simple security condition.
  – From the choice of $v'$, either $b' - b = \emptyset$ or $b' - b = \{ (s_2, o, r) \}$
  – If $b' - b = \emptyset$, then $\{ (s_2, o, r) \} \in b$, so $v = v'$, proving that $v'$ satisfies the simple security condition.
  – If $b' - b = \{ (s_2, o, r) \}$, because the get-read rule requires that $f_c(s)$ $dom f_o(o)$, an earlier result says that $v'$ satisfies the simple security condition.

Proof

• Consider the *-property.
  – Either $s_2 \in S_T$ or $f_c(s)$ $dom f_o(o)$ from the definition of get-read
  – If $s_2 \in S_T$, then $s_2$ is trusted, so *-property holds by definition of trusted and $S_T$.
  – If $f_c(s)$ $dom f_o(o)$, an earlier result says that $v'$ satisfies the simple security condition.
Proof

• Consider the discretionary security property.
  – Conditions in the get-read rule require \( r \in m[s, o] \) and either \( b' - b = \emptyset \) or \( b' - b = \{(s_2, o, r')\} \)
  – If \( b' - b = \emptyset \), then \( \{(s_2, o, r')\} \in b \), so \( v = v' \), proving that \( v' \) satisfies the simple security condition.
  – If \( b' - b = \{(s_2, o, r')\} \), then \( \{(s_2, o, r')\} \notin b \), an earlier result says that \( v' \) satisfies the ds-property.

give-read Rule

• Request \( r = (s_1, \text{give}, s_2, o, r') \)
  – \( s_1 \) gives (request to give) \( s_2 \) the (discretionary) right to read \( o \)
  – Rule: can be done if giver can alter parent of object
    • If object or parent is root of hierarchy, special authorization required
• Useful definitions
  – root\((o)\): root object of hierarchy \( h \) containing \( o \)
  – parent\((o)\): parent of \( o \) in \( h \) (so \( o \in h(\text{parent}(o)) \))
  – canallow\((s, o, v)\): \( s \) specially authorized to grant access when
    object or parent of object is root of hierarchy
  – \( m \leftarrow m[s, o] \): access control matrix \( m \) with \( r \) added to \( m[s, o] \)
give-read Rule

- Rule is $\rho_6(r, v)$:
  
  $\text{if } (r \neq \Delta(\rho_6)) \text{ then } \rho_6(r, v) = (i, v)$;
  
  $\text{else if } ([o \neq \text{root}(o) \text{ and parent}(o) \neq \text{root}(o) \text{ and parent}(o) 
  \in b(s_1:w)] \text{ or } [\text{parent}(o) = \text{root}(o) \text{ and canallow}(s_1, o, v)] \text{ or } [o = \text{root}(o) \text{ and canallow}(s_1, o, v)] \text{ then } \rho_6(r, v) = (y, (b, m[x, y] \leftarrow r, f, h));$

  $\text{else } \rho_1(r, v) = (n, v);$  

Security of Rule

- The give-read rule preserves the simple security condition, the *-property, and the ds-property
  
  - Proof: Let $v$ satisfy all conditions. Let $\rho_1(r, v) = (d, v')$. If $v' = v$, result is trivial. So let $v' = (b, m[x, y] \leftarrow r, f, h), b' = b, f' = f, m[x, y] = m'[x, y]$ for all $x \in S$ and $y \in O$ such that $x \neq s$ and $y \neq o$, and $m[s, o] \subseteq m'[s, o]$. So, by earlier result, $v'$ satisfies the simple security condition, the *-property, and the ds-property.
Principle of Tranquility

• Raising object’s security level
  – Information once available to some subjects is no longer available
  – Usually assume information has already been accessed, so this does nothing

• Lowering object’s security level
  – The declassification problem
  – Essentially, a “write down” violating *-property
  – Solution: define set of trusted subjects that sanitize or remove sensitive information before security level lowered

Types of Tranquility

• Strong Tranquility
  – The clearances of subjects, and the classifications of objects, do not change during the lifetime of the system

• Weak Tranquility
  – The clearances of subjects, and the classifications of objects, do not change in a way that violates the simple security condition or the *-property during the lifetime of the system
Example

• DG/UX System
  – Only a trusted user (security administrator) can lower object’s security level
  – In general, process MAC labels cannot change
    • If a user wants a new MAC label, needs to initiate new process
    • Cumbersome, so user can be designated as able to change process MAC label within a specified range

Controversy

• McLean:
  – “value of the BST is much overrated since there is a great deal more to security than it captures. Further, what is captured by the BST is so trivial that it is hard to imagine a realistic security model for which it does not hold.”
  – Basis: given assumptions known to be non-secure, BST can prove a non-secure system to be secure
†-Property

- State \((b, m, f, h)\) satisfies the †-property iff for each \(s \in S\) the following hold:
  1. \(b(s; a) \neq \emptyset \Rightarrow \forall o \in b(s; a) \left[ f_o(s) \text{ dom } f_o(o) \right]\)
  2. \(b(s; w) \neq \emptyset \Rightarrow \forall o \in b(s; w) \left[ f_o(s) = f_w(s) \right]\)
  3. \(b(s; r) \neq \emptyset \Rightarrow \forall o \in b(s; r) \left[ f_o(s) \text{ dom } f_o(o) \right]\)

- Idea: for writing, subject dominates object; for reading, subject also dominates object

- Differs from *-property in that the mandatory condition for writing is reversed
  - For *-property, it’s object dominates subject

Analogues

The following two theorems can be proved

- \(\Sigma(R, D, W, z_0)\) satisfies the †-property relative to \(S^\prime \subseteq S\) for any secure state \(z_0\) iff for every action \((r, d, (b, m, f, h), (b', m', f', h'))\), \(W\) satisfies the following for every \(s \in S^\prime\)
  - Every \((s, o, p) \in b - b'\) satisfies the †-property relative to \(S^\prime\)
  - Every \((s, o, p) \in b'\) that does not satisfy the †-property relative to \(S^\prime\) is not in \(b\)

- \(\Sigma(R, D, W, z_0)\) is a secure system if \(z_0\) is a secure state and \(W\) satisfies the conditions for the simple security condition, the †-property, and the discretionary security property.
Problem

• This system is *clearly* non-secure!
  – Information flows from higher to lower because of the †-property

Discussion

• Role of Basic Security Theorem is to demonstrate that rules preserve security
• Key question: what is security?
  – Bell-LaPadula defines it in terms of 3 properties (simple security condition, ‡-property, discretionary security property)
  – Theorems are assertions about these properties
  – Rules describe changes to a particular system instantiating the model
  – Showing system is secure requires proving rules preserve these 3 properties
Rules and Model

- Nature of rules is irrelevant to model
- Model treats “security” as axiomatic
- Policy defines “security”
  - This instantiates the model
  - Policy reflects the requirements of the systems
- McLean’s definition differs from Bell-LaPadula
  - … and is not suitable for a confidentiality policy
- Analysts cannot prove “security” definition is appropriate through the model

System Z

- System supporting weak tranquility
- On any request, system downgrades all subjects and objects to lowest level and adds the requested access permission
  - Let initial state satisfy all 3 properties
  - Successive states also satisfy all 3 properties
- Clearly not secure
  - On first request, everyone can read everything
Reformulation of Secure Action

- Given state that satisfies the 3 properties, the action transforms the system into a state that satisfies these properties and eliminates any accesses present in the transformed state that would violate the property in the initial state, then the action is secure.
- BST holds with these modified versions of the 3 properties.

Reconsider System Z

- Initial state has subject $s$, object $o$, $C = \{\text{High, Low}\}$, and $K = \{\text{All}\}$. Take $f_c(s) = (\text{Low, } \{\text{All}\})$, $f_o(o) = (\text{High, } \{\text{All}\})$, $m[s,o] = \{ w \}$, and $b = \{ (s, o, w) \}$.
- $s$ requests $r$ access to $o$.
- Now $f_o^{'}(o) = (\text{Low, } \{\text{All}\})$, $(s, o, r) \in b^{'}$, and $m[s,o] = \{r, w\}$.
Non-Secure System Z

- As \((s, o, r) \in b^\prime - b\) and \(f_o(o) \text{dom} f_c(s)\), access added that was illegal in previous state
  - Under the new version of the Basic Security Theorem, System Z is not secure
  - Under the old version of the Basic Security Theorem, as \(f_c^\prime(s) = f_o^\prime(o)\), System Z is secure

Response: What Is Modeling?

- Two types of models
  1. Abstract physical phenomenon to fundamental properties
  2. Begin with axioms and construct a structure to examine the effects of those axioms
- Bell-LaPadula Model developed as a model in the first sense
  - McLean assumes it was developed as a model in the second sense
Reconciling System Z

• Different definitions of security create different results
  – Under one (original definition in Bell-LaPadula Model), System Z is secure
  – Under other (McLean’s definition), System Z is not secure

Key Points

• Confidentiality models restrict flow of information
• Bell-LaPadula models multilevel security
  – Cornerstone of much work in computer security
• Controversy over meaning of security
  – Different definitions produce different results
Overview of Integrity

• Requirements
  – Very different than confidentiality policies
• Biba’s models
  – Low-Water-Mark policy
  – Ring policy
  – Strict Integrity policy
• Lipner’s model
  – Combines Bell-LaPadula, Biba
• Clark-Wilson model

Requirements of Policies

1. Users will not write their own programs, but will use existing production programs and databases.
2. Programmers will develop and test programs on a nonproduction system; if they need access to actual data, they will be given production data via a special process, but will use it on their development system.
3. A special process must be followed to install a program from the development system onto the production system.
4. The special process in requirement 3 must be controlled and audited.
5. The managers and auditors must have access to both the system state and the system logs that are generated.